12

NW

Semiannual Technical Report
Covering the period 12 May 1976 through 12 November 1976

INTERACTIVE AIDS FOR CARTOGRAPHY AND PHOTO INTERPRETATION

By: HARRY G. BARROW
Principal Investigator
(415) 326-6200, Ext. 5089

Sponsored by

DEFENSE ADVANCED RESEARCH PROJECTS AGENCY
ARLINGTON, VIRGINIA 22209

CONTRACT DAAG29-76-C-0057 new
ARPA Order No. 2894-5
Program Element Code 61101E

STANFORD RESEARCH INSTITUTE
Menlo Park, California 94025 · U.S.A.

BEST
AVAILABLE COPY

STANFORD RESEARCH INSTITUTE
Menlo Park, California 94025 · U.S.A.

9

Semiannual Technical Report.
Covering the period 12 May ~~1976 through~~ 12 November 1976,
Stanford Research Institute Project 5300

11

November 1976

12 52p.

6

INTERACTIVE AIDS FOR CARTOGRAPHY AND PHOTO INTERPRETATION.

10

Submitted by:
HARRY G. BARROW
Principal Investigator
(415) 326-6200, Ext. 5089

15

CONTRACT DAAG29-76-C-0057 new
ARPA Order ~~No.~~ 2894-5
Program Element Code 61101E

Effective Date: 12 May 1976
Expiration Date: 9 October 1977
Amount of Contract: $521,363

Prepared for
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY
ARLINGTON, VIRGINIA 22209

Approved by:

PETER E. HART, *Associate Director*
Artificial Intelligence Center

EARLE D. JONES, *Associate Executive Director*
Information Science and Engineering Division

ABSTRACT

This report describes the work performed during the first six months of our project on Image Understanding. The central scientific goal of the research program is to investigate and develop ways in which diverse sources of knowledge may be brought to bear on the problem of interpreting images. The research is focused on the specific problems entailed in interpreting aerial photographs for cartographic or intelligence purposes, with a view to the eventual development of a collaborative aid to the cartographer or photo interpreter. A key concept is the use of a generalized digital map to guide the process of image interpretation.

CONTENTS

ILLUSTRATIONS

ACKNOWLEDGMENTS

The work described in this report was carried out by H. G. Barrow, R. Bolles, T. D. Garvey, J. Kremers, J. M. Tenenbaum, and H. C. Wolf of the Artificial Intelligence Center, Stanford Research Institute.

The research was supported by the Defense Advanced Research Projects Agency of the Department of Defense and was monitored by the U.S. Army Research Office under Contract No. DAAG29-76-C-0057.

I INTRODUCTION AND OVERVIEW

A. Research Program and Plan

The central scientific goal of our research program is to investigate and develop ways in which diverse sources of knowledge may be brought to bear on the problem of interpreting images. The research is focused on the specific problems entailed in interpreting aerial photographs for cartographic or intelligence purposes, with a view to the eventual development of a collaborative aid to the cartographer or photo interpreter. A key concept is the use of a generalized digital map to guide the process of image interpretation.

B. Overview of Work Carried Out

The work of the past six months has adhered to the research plan laid down in the contract proposal. Roughly half of our effort has been concerned with developing a skeleton integrated system within which experimentation could be readily carried out, and the remaining half has been directed to investigating a variety of tasks, using the system with real aerial images. In practice, system development has proceeded better than expected and we have been able to devote time to in depth investigation of some tasks.

The zero-order skeleton system is described in [1]. The main extensions that have been made to this system in the past six months are:

* The capability of handling three dimensional information.
* Modification of the basic representation of the map.

The tasks investigated during the past six months have been :-

* Detecting and counting boxcars in a railroad yard,

* Tracing roads interactively for mapmaking.

In addition, we have begun to study the general problem of representing many different varieties of knowledge coherently, and to study the semantics of understanding aerial imagery.

II WORK PERFORMED DURING THE PAST SIX MONTHS

A. Image Calibration

1. Overview

A critical step in using map (or world) data to aid in picture interpretation is the transformation of the data from world coordinateS to picture coordinates. This is traditionally accomplished by using a calibrated camera position to create a function that accepts world coordinates and returns image coordinates. The calibration of the camera is the key to creating this function. We have written a system that calibrates the camera, using control points for which both world coordinates and image coordinates are known. In addition, we have produced programs that allow the user to interactively specify these points.

2. Techniques

The calibration routines attempt to determine the altitude and location of the camera in the world frame, and the heading, pitch, and roll angle deviations from straight and level flight in a northerly direction. Other camera parameters, including focal length, aspect ratio, and scale factor may also be computed.

The straightforward calibration procedure estimates the camera parameters based on the input set of control points. This estimate is used to create a camera transformation function (or camera transform) from which both the location and location error of the projected control points in the image is computed. A steepest descent optimization algorithm is used to minimize the error resulting from performing the transformation, and thus, to optimize the camera calibration.

The current procedure for determining appropriate control points for the calibration routines is to project a set of predetermined control points into a displayed version of the image using a rough, preliminary camera transform. This rough transform is derived from world coordinates of the corners of the image. The correct location of these points is then indicated by the user, and the resulting corrected data is passed to the calibration routines for creation of a final transform.

In the near future, we will attempt to reduce user input in the determination of the correct location of the control points. This will be done by associating with each control point a subwindow ("chip") from a previously calibrated image. This chip will contain the control feature (which might be, for example, an intersection of two roads) associated with the point. These image chips will be correlated within the new image in an attempt to locate the point accurately and automatically.

B. Boxcar Counting

1. Overview

A common type of task performed by photo interpreters is classifying and counting large numbers of similar objects, such as oil storage tanks, aircraft or boxcars. Such tasks are tedious and demanding, but they do not necessarily require a high degree of skill. They are thus suitable candidates for automation or semi-automation.

We considered several specific counting tasks and selected counting boxcars in a railroad yard as the domain for our exploratory studies. Our main reason for choosing this task was that it is also a monitoring task: It may be necessary to count the cars in a particular yard on many occasions. In this case, we can exploit our a priori knowledge of the layout of the rails, together with knowledge of different types of boxcars to facilitate performance.

Our aim in this exploratory study was not to write a program that could perform as well as a human photo interpreter; we had the more modest aim of finding out what might be done to provide an interactive aid that could make the photo interpreter's task easier.

2. Techniques

For the purposes of this study, we assumed that we have already made a digital map of the rail yard and have calibrated the image in question so that the map and image can be put into correspondence accurately. The effective dimensionality of the problem of searching for boxcars is thus reduced from two or three dimensions to one; we need only scan along the lines indicated by the map in searching for cars.

The system operates by scanning a length of track indicated by the user, looking for a train of cars. The train is then scanned, looking for the divisions between cars. When the train has been segmented into individual cars, the results are displayed to the user who may correct errors, if they exist. The counts of cars classified by length may then be printed. A future system might analyze the image of each car in detail to classify it more appropriately.

We shall now describe the algorithms used in more detail: *Since detecting the start and end of a train is a special case* of box car end detection, we shall describe the train segmentation process first.

The first step is to extract from the picture a narrow strip aligned along the track in question, with width just sufficient to contain the image of a train. The strip is sliced perpendicular to the track and the mean brightness is computed for that portion of each slice that would be contained within the train (see Figure 1). The problem is thus reduced to segmenting a one dimensional sequence of means.

Two statistical operators are then scanned along the sequence of means. The operators attempt to detect gaps between cars based on a

simple model; there are constraints on the dimensions of cars and gaps, and cars and gaps usually are different in brightness. Each operator looks at two sections of the sequence, one section just shorter than the minimum car length, the other just shorter than the minimum gap length; the two sections are separated by a space to allow for resolution limitations (see Figure 2).

For each section, the mean and standard deviation of the mean brightnesses is computed, and a simple test is made to determine the significance of any difference between the section means, in terms of a number of standard deviations. The operator is thus, to some extent, self scaling against image noise, illumination, markings, and so forth. When one section is squarely on a car and the other on a gap, the difference may be 50 to 70 standard deviations in good cases.

Each operator is asymmetric, and the two are mirror images of each other. One can be considered to be looking for the start of an intercar gap, and the other for the end of the gap. Figure 3 shows a hypothetical train, and Figure 4 shows the output from the two operators.

In the next step, a significance threshold (about three standard deviations) is applied to the operator outputs, and the local maxima of the above-threshold regions are found. Thus a sequence of events is extracted, each event being significant peak response of one of the operators. The threshold is chosen so that few true events are missed, and some false alarms are found; These will be rejected by context later. Figure 5 shows the events detected in the example. Note that where a dark gap occurs, the two operators yield events that both have negative sign, are close together, and are in the right order ("startgap" - "endgap"). Where the gap is unclear and only a brightness transition can be seen, the operators yield two events of opposite sign simultaneously. Similar behavior is observed at the ends of the train.

The sequence of events is then scanned and interpreted. For explanation purposes, we shall describe a simpler algorithm than that

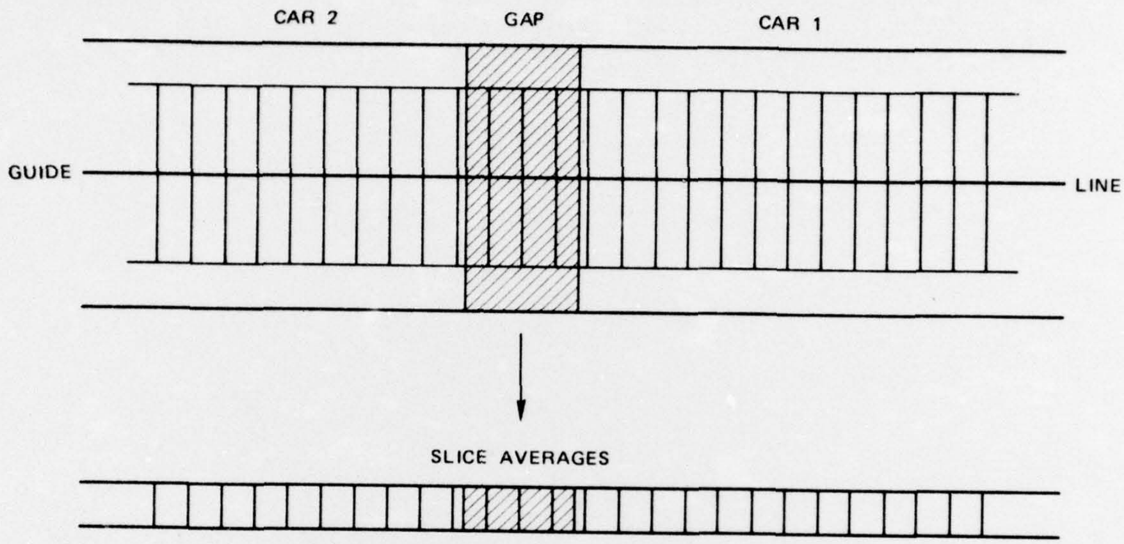


FIGURE 1 A PORTION OF A TRAIN

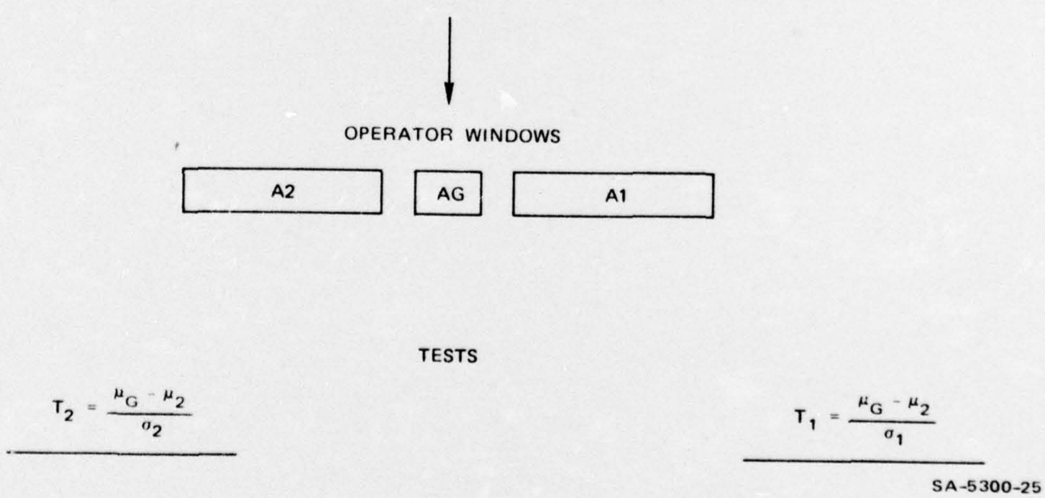


FIGURE 2 GAP DETECTION OPERATORS

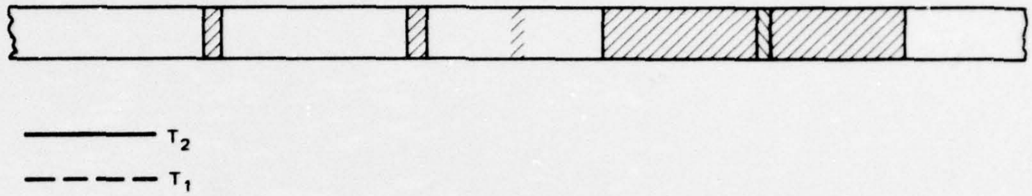


FIGURE 3 A HYPOTHETICAL TRAIN

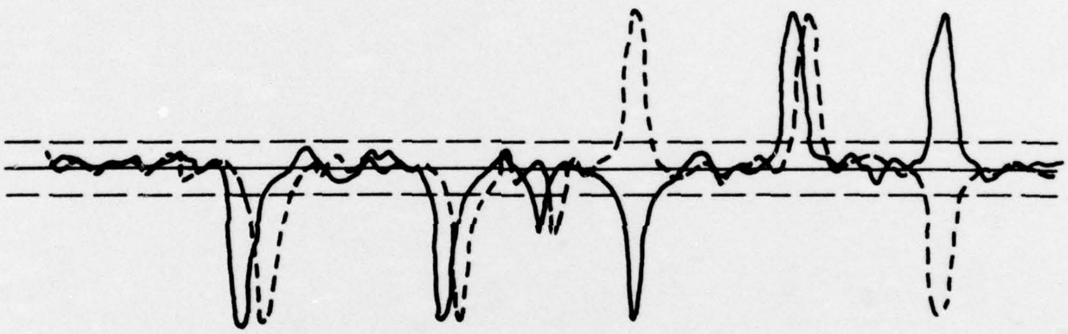
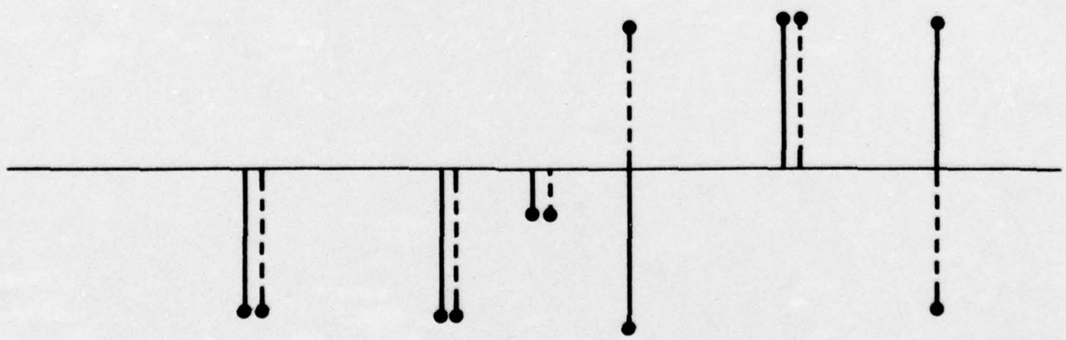


FIGURE 4 RESULTS OF THE GAP DETECTORS



SA-5300-24

FIGURE 5 DETECTED EVENTS

actually used. At a given point in the scan, we have a hypothesis about the correct context for that point: either boxcar, gap, or empty track. The next event is interpreted in the light of the current context. If the next event occurs too soon, for example a start of gap event closer than the minimum possible length of a boxcar, it can probably be rejected as spurious. A transition in brightness could simply indicate a hard-to-see gap, or the end of the train. The end of the train can be ruled out if the prospective length of track has characteristics different from those of an example of empty track provided by the user.

The net result of the interpretation algorithm is a segmentation of the sequence of data into lengths corresponding to boxcars, gaps, or track. The program displays the results to the user by marking each boxcar with a red dot. The user can then check the interpretation for obvious errors, and can interactively make corrections by pointing to the car in question and indicating its true end points with the track ball cursor. The aim here, we reiterate, is not the unattainable goal of fully automatic counting, but easing the photo interpreter's task by allowing the system to perform most of the task, with a limited amount of guidance and correction from the user.

When the user has had the opportunity of correcting the system's mistakes, if any, the count of boxcars classified by length is printed.

3. Performance

The program has been tested on images provided to us by 544th Aerospace Reconnaissance Technical Wing, Strategic Air Command. Figure 6 shows a portion of a reconnaissance photo depicting San Francisco railroad yard, digitized at a resolution equivalent to 10000 x 10000 pixels over the original image, and a digital zoom into an area of interest. In the close-up picture a guide-line representing a portion of track is superimposed over a train, together with a window provided by the user as an example of empty track. Figure 7 shows the boxcars

found by the program; the train has been correctly interpreted and all box cars have been correctly delimited and counted. Figure 8 shows a more difficult example. In this case, the program missed one gap between boxcars, and the user interactively corrected the interpretation. The classification and counting on the corrected data was performed automatically.

We have not yet made enough tests of the system to accurately assess its performance, but preliminary subjective assessments as follows. The system performs well on clean data, giving correct segmentation and counts when gaps between cars are well defined. It makes relatively few errors, of the order of a few percent, in cases where the gaps are hard to see. It is not known how well humans perform in similar tasks, but in many cases accuracy to a few per cent may be acceptable. The system has not been tested outside the domain of boxcars.

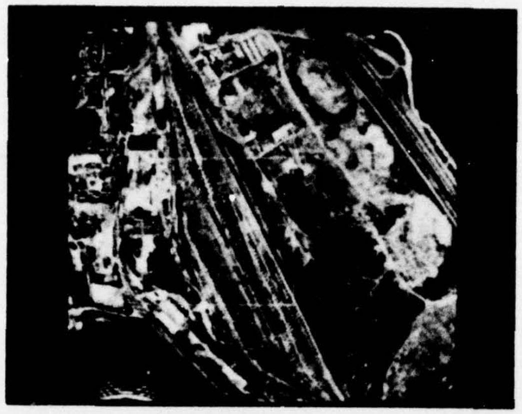
Clearly, to reach a similar level of performance in cases where appearances of cars are not so predictable requires further work. The results obtained so far have been sufficiently encouraging for us to consider the domain of railroad yard analysis as a possible candidate for an expert subsystem.

C. Road Tracing

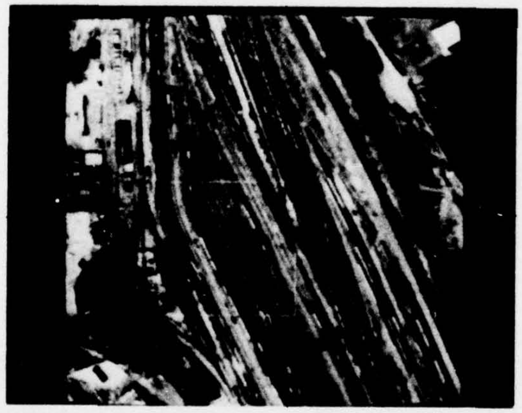
1. Overview

A major current problem in mapmaking, as currently practised by DMA, is that of tracing linear cartographic features, such as roads, rivers, or railroads, in aerial imagery. Most of the steps of the map making process have been automated, but feature extraction, and linear feature extraction in particular, are still performed manually. There is a definite need for automation of this remaining bottleneck.

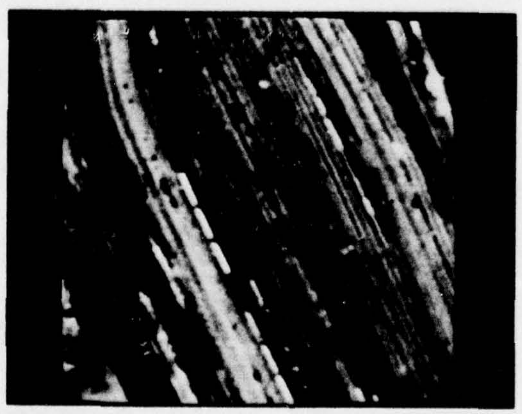
Earlier attempts at tracing linear features have met with little success, largely because of the lack of exploitation of knowledge of the problem. We felt that the techniques of Image Understanding



(a)



(b)



(c)

FIGURE 6 SAN FRANCISCO RAILROAD YARD

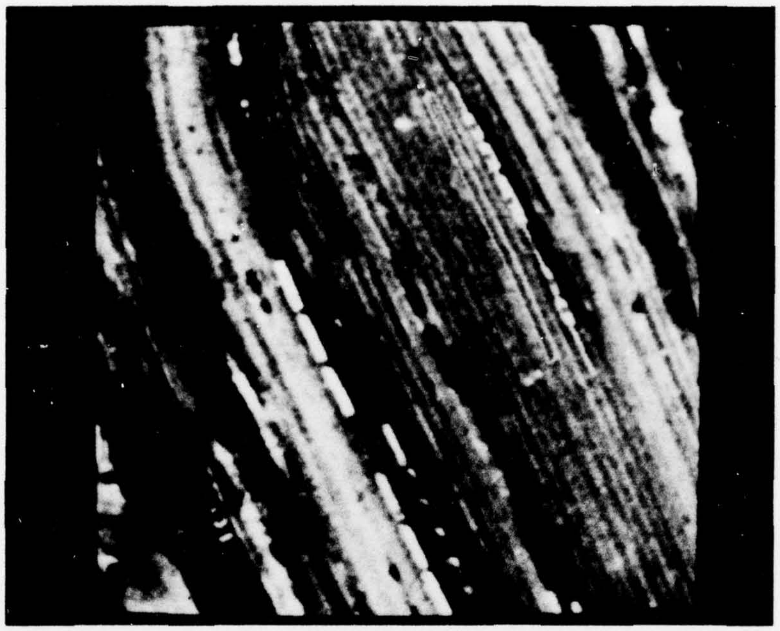


FIGURE 7 RESULT OF BOX CAR COUNTING

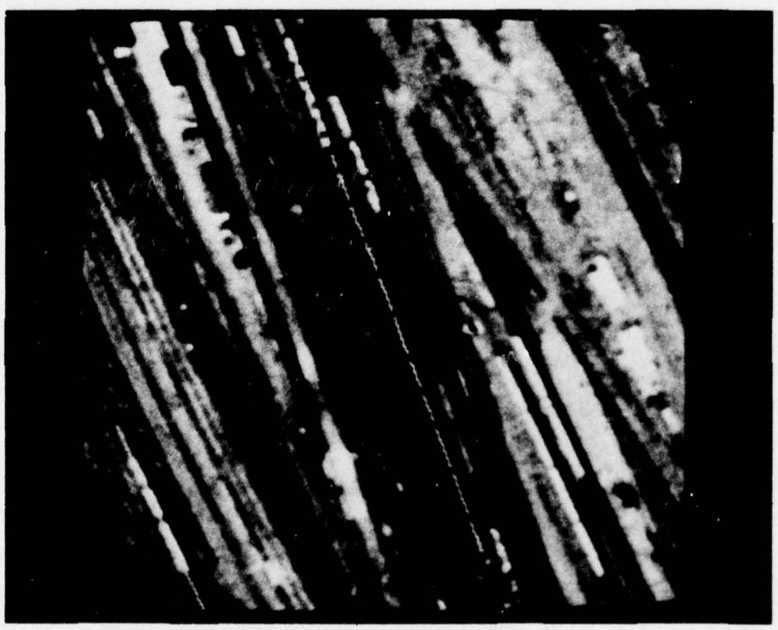


FIGURE 8 INTERACTIVELY CORRECTED BOX CAR COUNTING

could make useful impact on this outstanding problem, and we have made a preliminary investigation. The philosophy we adopt is, once more, that a fully automatic feature extraction is beyond the current state of the art, but a semiautomated, interactive approach can be very effective. We have thus been considering situations in which a user provides an approximate course of the linear feature in question, and the system uses this guideline to trace the feature accurately.

A system that can exploit approximate information to trace features also has application in photo interpretation. If we already have a map of an area, and we know approximately the parameters of the camera when the photo was taken, the approximate location of known linear features in the photo can be predicted. The prediction can be used as a guideline for tracing features accurately. Thus, a substantial amount of interpretation of the image may be performed automatically.

2. Exploration

We began with the intent of drawing upon established techniques for line finding and tracing, and undertook a broad range of experiments to determine the best current techniques. We found, however, that all the established techniques were inappropriate in some way, chiefly because they were not "tuned" to aerial imagery, and consequently had poor performance. We were therefore obliged to develop techniques specific to tracing linear cartographic features, such as roads.

There are three main stages in the process of tracing linear features:

* Detection--Finding points through which a linear feature passes, together with some measure of confidence for each point, by a very local test.
* Enhancement--Modifying the confidences in the light of local context: For example, increasing confidence for a point between two points of high confidence.

* Tracing--Finding the best continuous path through the image that links points of high confidence: a global operation.

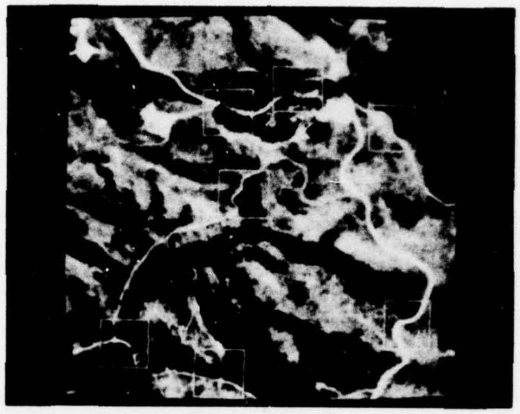
These stages may have much in common: the distinction is made here mainly for explanation purposes.

We provided ourselves with a set of 13 subpictures, each of 32 x 32 pixels, covering a variety of road situations, including rural, suburban, and urban areas, for use as experimental test cases. The pictures from which they were selected, and the subimages themselves, are shown in Figure 9 to Figure 11. From the subimages, it can be seen how difficult it may be to detect the presence of a road from local evidence alone, especially in urban areas, where the subimage looks like random noise. Even in rural areas where things are clearer, the road may not always appear as an ideal line; one edge may disappear, because of shading, or the road may be partially or totally obscured locally by trees.

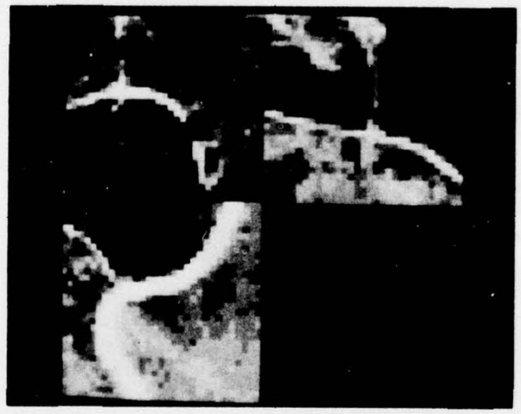
a. Detection

A simple local approximation to a road in a photograph is a line segment that differs in brightness from the background, where line and background are each assumed to be of uniform brightness. We here assume that the details of the appearance of the road are invisible under the resolution of the image. Various operators for detecting line-like features have been developed. Most of them assume a step-shaped intensity profile, form a best estimate of the parameters (such as mean brightness), and test whether parameters are significantly different. The Hueckel operator [2] is a much quoted and used operator that can estimate many parameters, including position, orientation and width, of the line. It was an obvious prime candidate for line segment detection.

The results of experiments with the Hueckel operator were, however, disappointing. It appeared to perform well only on very strong lines in real aerial images, and performed poorly in congested, textured areas such as urban areas. (See Figure 12.)



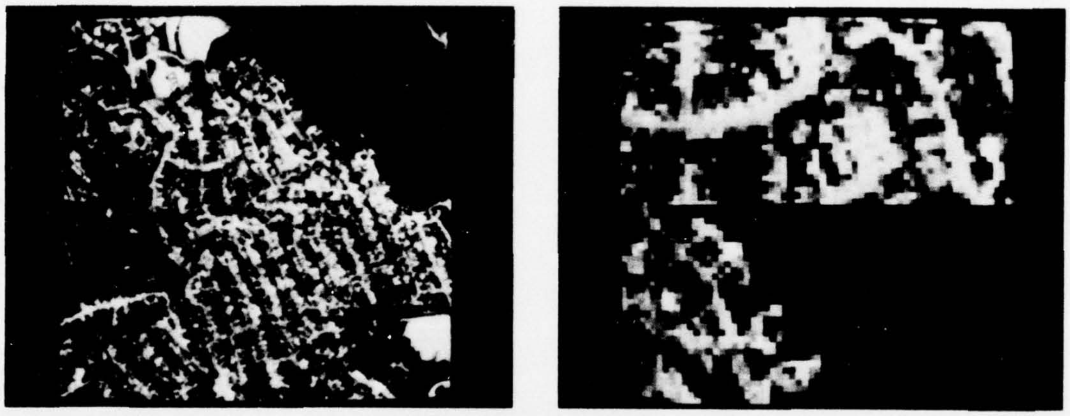
(a)



(b)

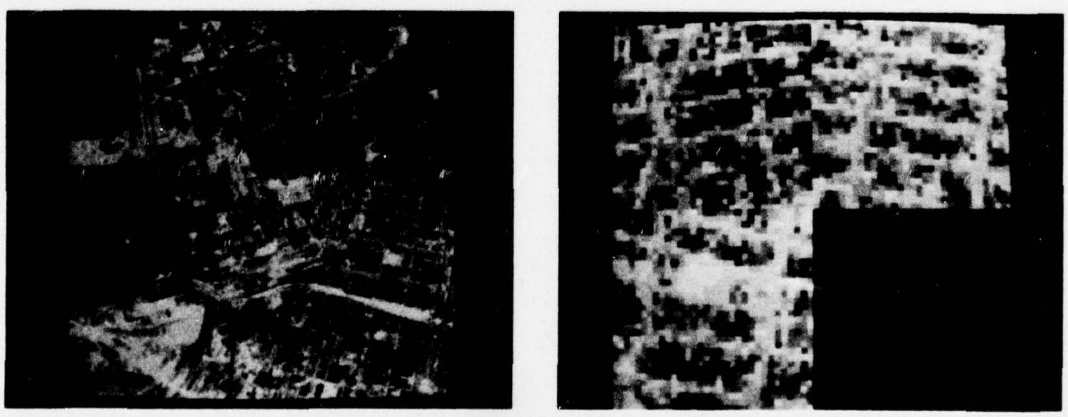
(c)

FIGURE 9 RURAL TEST SUBIMAGES



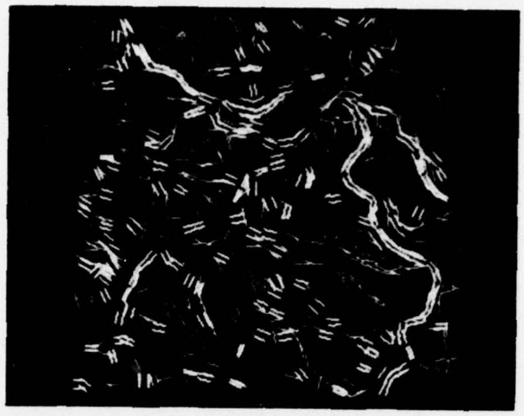
(a) (b)

FIGURE 10 SUBURBAN TEST SUBIMAGES

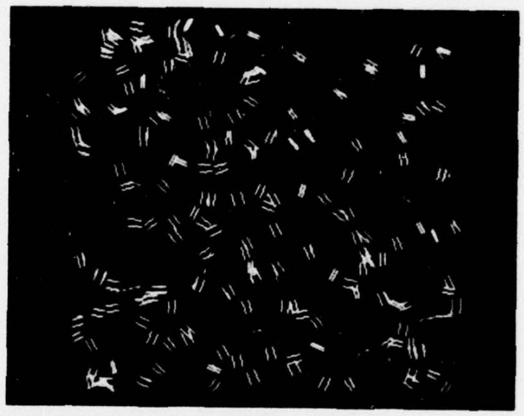


(a) (b)

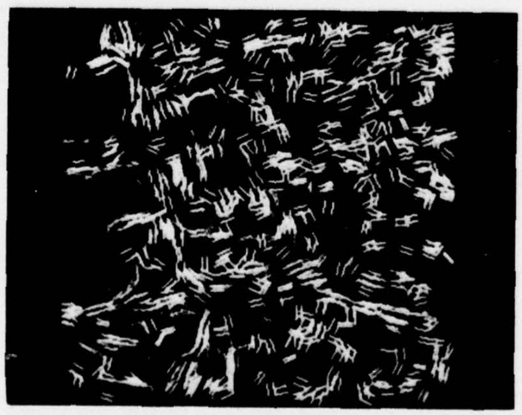
FIGURE 11 URBAN TEST SUBIMAGES



(a)



(b)



(c)

FIGURE 12 OUTPUT OF HUECKEL LINE/EDGE OPERATOR

Experiments with a simple gradient operator, the Sobel operator [3], showed evidence for significant brightness gradient even where edges or lines were not detected by the Hueckel operator. (See Figure 13.) Apparently the assumptions of a locally uniform line on a uniform background, upon which the Hueckel operator is based, were inappropriate. The human eye, on the other hand, seems to be detecting much more local events, like high gradient, and responding to chains of such events.

For our next experiment, we used a Sobel operator to measure brightness gradient for each point in the image and then used a heuristic search technique, described below, to trace one side of the road. The search routine was given start and finish points and was required to find a path between them such that a scoring function was maximized. Two main scoring functions were considered: total brightness gradient minus path length, and average brightness gradient over path. Neither was satisfactory: the former had a tendency to take short cuts, whereas the latter tended to wander, and even to loop. Both scoring functions led to much branching in the search. Using a heuristic estimate of the score of the final path to the goal (instead of just using the score for the path so far found) reduced the amount of search in each case, but only by about 20%.

A major drawback of tracing the paths with maximum gradient score was that edges near the road could easily distract the tracing process, even if they were not immediately adjacent to the road. Using a nonlinear function of the gradient, such as its square, somewhat reduced the degree of distraction, but results were still not satisfactory.

We concluded that it was necessary to design a special operator for detecting road fragments locally to avoid the pitfalls we had encountered with the other operators. In particular, assumptions of uniformity of background are invalid, and detecting only one side of the road, as with an edge operator, ignored potentially useful structural

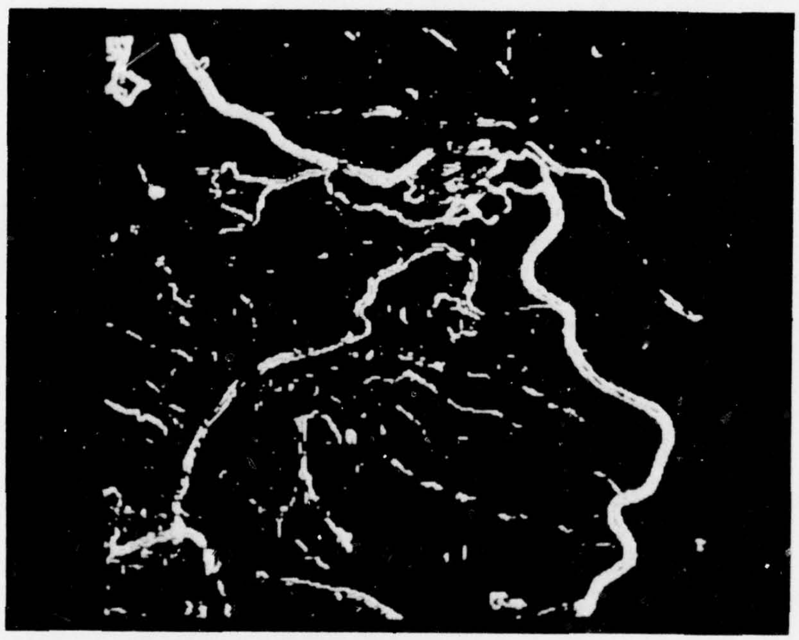


FIGURE 13 OUTPUT OF SOBEL GRADIENT OPERATOR

information. Accordingly, we developed a road operator that incorporates more knowledge about the appearance of roads than a general-purpose line detector.

We began by first considering the appearance of roads in general: A road is an area that is narrow in proportion to its length, with a restricted range of widths, and usually curves gently or is locally straight. It is substantially homogeneous, so its visual characteristics vary only slowly along its length, although they may differ considerably from those of the areas on either side.

With this description in mind, we developed an operator for road detection. The detector looks only at a small area of the image, one that is chosen to be large enough to span the road and some terrain on each side, but that is small enough for us to consider the road fragment as essentially straight. We look at a few points that we expect to lie within the road and a few on either side that we expect not to lie on the road. We space these test points sufficiently to allow for different widths of road (over a small range) or possible small curvatures. The test we make is composed of two parts: a test for difference between points on the road and points off it--we assume that usually there will be an observable difference between road and background--and a test for approximate uniformity within the road.

At the scale and resolution of imagery with which we are working, an appropriate size for the operator is 5 x 5 pixels. We attempt to detect roads in one of a few quantized orientations--for this size of operator, four orientations is appropriate. Two of the family of operators are shown in Figure 14; the other two are similar but rotated through 90 degrees. Three points in a row--a1, a2, and a3--are assumed to lie on the road, and three points on each side--b1, b2, b3, and c1, c2, c3--are assumed to lie on the background.

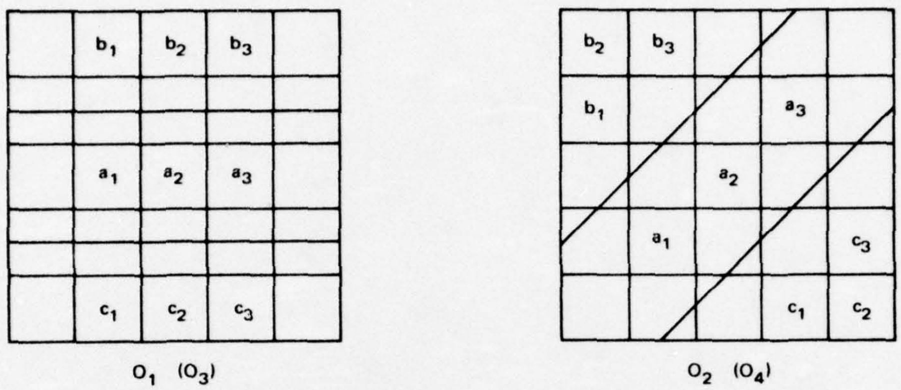
To test for difference from the background terrain, we take pairs of points, one on the road and one off, and measure the difference in their brightnesses, (ai - bi) or (ai - ci). We are

interested in determining a score for the set of six differences, and we are prepared to tolerate some of the differences being low, provided the remainder are high. For this reason we use a sum of scores for the individual differences. The score for a difference is given by a function, F, shown in Figure 15. The function is nonlinear to reflect the fact that when a difference is greater than a certain value, its actual magnitude is unimportant; below that value, the score depends on the magnitude. It is also symmetric if we do not know whether the road is lighter or darker than the background but asymmetric when we do have such knowledge. In our examples, the road was known to be lighter. The function F is intended to reflect the likelihood that the observed difference is not significant.

To test for uniformity of the road, we take adjacent pairs of points along the road and measure the differences in their brightnesses, (a1 - a2) and (a2 - a3). In this case, we wish to determine a score for the two differences, tolerating low difference values but not high ones. We therefore compute a product of scores, the score for each difference being given by a function, G, shown in Figure 16. Note that G gives uniform high scores to low differences, and uniform low scores to high differences, with a continuous transition in the range of intermediate difference values. The function is also symmetric, since the sign of the difference is not important. The function G is intended to be an approximation to the likelihood that the observed difference is not significant.

The overall operator score is given by the ratio of the two terms, the road uniformity score divided by the difference-from-background score. The operator score varies from 0 (no road) to 1 (good road).

The results of applying the operator to the selected test windows is shown in Figure 17. To display results intelligibly, the operators for the four orientations were applied at each point, and the highest score was compared with a fixed threshold to determine whether a



$$\text{SCORE} = \prod_{i=1}^{2} G(a_i - a_{iH}) \Big/ \sum_{i=1}^{3} F(a_i - b_i) + F(a_i - a)$$

FIGURE 14 WINDOWS FOR THE ROAD OPERATORS

WHERE:

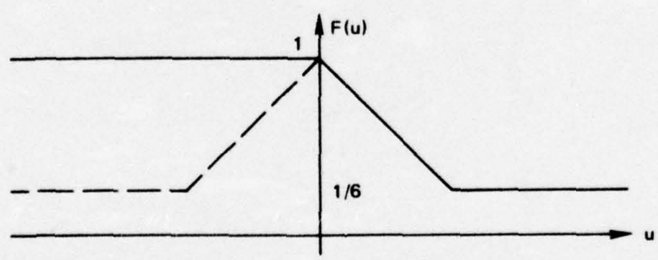
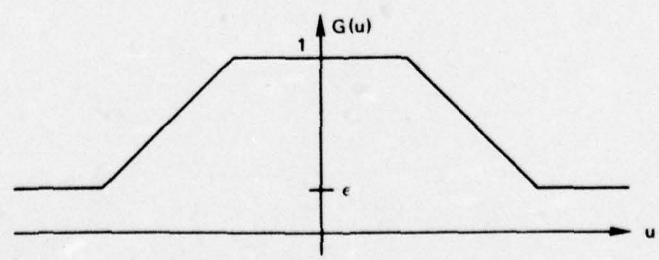


FIGURE 15 ROAD EDGE SCORING FUNCTION



SA-5300-26

FIGURE 16 ROAD UNIFORMITY SCORING FUNCTION

road point should be displayed. (In normal use the scores are not thresholded.)

b. Enhancement

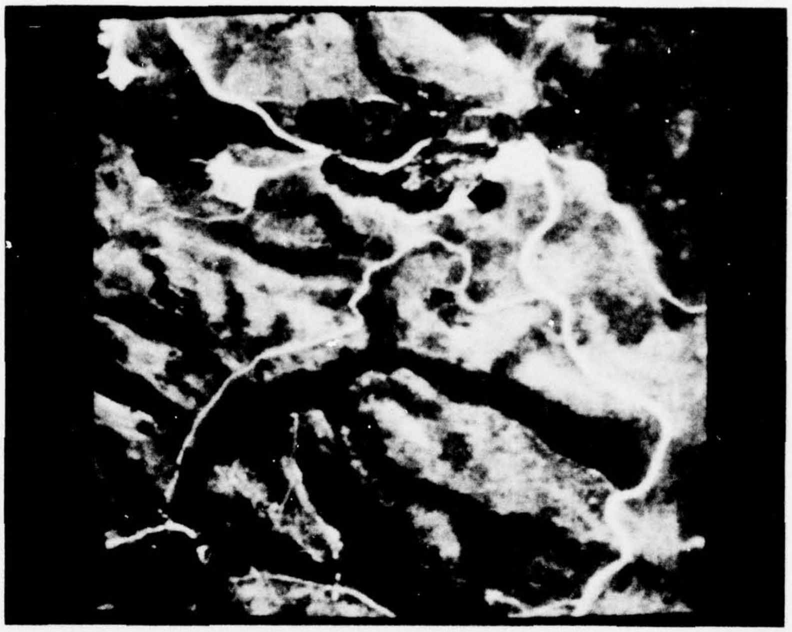
We investigated the possibility of applying relaxation methods * to the enhancement of the road detecting operator's output. The key idea here is that for a given point in the image, we consider the operator output in the light of the surrounding local context. If the context indicates that a road actually passes through the point, but the operator happened to produce a low score, the score can be increased. Conversely, if the context indicates that no road passes through the point, a high score can be discounted as an accident, and decreased. Since the modified score depends on the neighboring (modified) operator scores, the enhancement process is iterative--or pseudo-parallel.

We experimented with a variety of ways of modifying scores. The common underlying principle is that segments of road that are aligned and adjacent support each other and mutually increase their scores; nonaligned adjacent segments contradict each other and mutually decrease their scores.

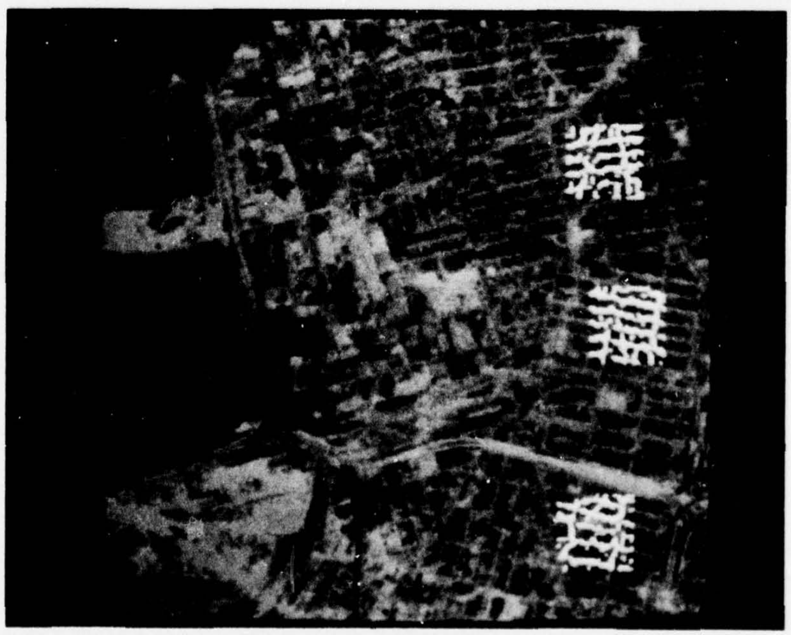
We found that taking a simple linear combination of supporting increments and contradicting decrements to modify a point score, while yielding some improvements, did not achieve the degree of enhancement we sought. A better approach was to look at a small neighborhood (3 x 3 points or 5 x 5 points) of the point in question, find the best scoring local path through the point in question, and update the point score solely on the basis of the best path. Figure 18 shows the original response of the road operator on a test image and Figure 19 shows the result of iterating this enhancement process a few times. Note that gaps have been filled, and ambiguities and noise have been reduced. However, we have not yet determined the rules governing

----

* Basic research into relaxation methods for scene analysis is being carried out concurrently, supported by a grant from NSF



(a)



(b)

FIGURE 17 OUTPUT OF THE ROAD OPERATOR

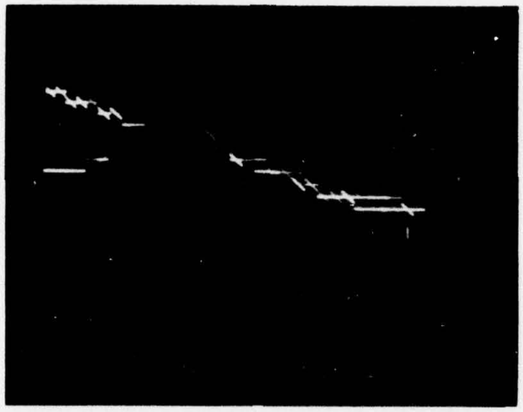


FIGURE 18 OUTPUT OF THE ROAD OPERATOR OVER A TEST WINDOW

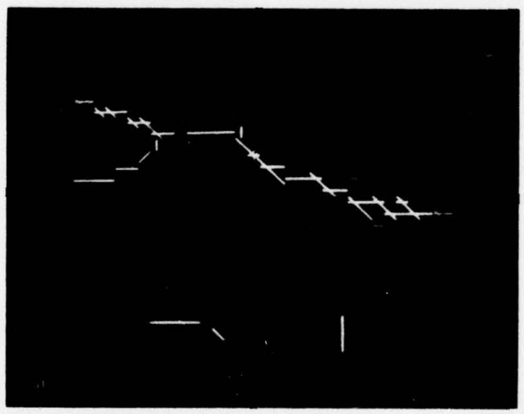


FIGURE 19 ENHANCEMENT AFTER 2 ITERATIONS

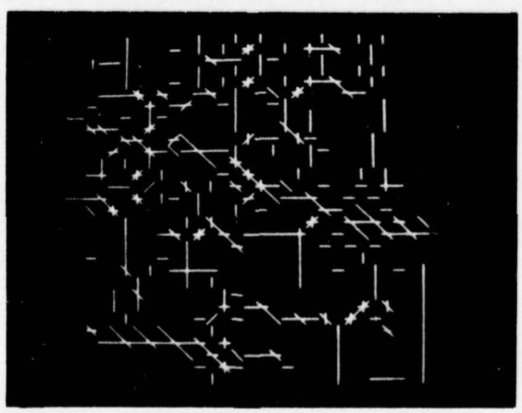


FIGURE 20 ENHANCEMENT AFTER 15 ITERATIONS

We used two principal measures of path cost: first, the integral of the step costs along the path, taking direction into account; and second, the average of the step costs along the path, also taking direction into account. The step cost of moving from one point to a neighboring point was determined by taking the average of the operator scores of the two neighboring points for the direction of the step, dividing by the length of the step, and subtracting it from unity. We thus have a measure that varies between 0 and 1 and is not direction sensitive.

As with the edge-tracing experiments described earlier, when the simple cost integral was used in the evaluation function, the algorithm had a tendency to take shortcuts through areas of moderate cost. When the average path cost was used, the algorithm had a tendency to wander. Accordingly, we tried using the integral of the square of step cost, to reduce shortcutting tendencies while maintaining goal-directedness. We found it gave good results in tracing, even in areas in which the road was hard to discern.

We also found that the A* algorithm was somewhat slow and was considering almost all points in the subimage under consideration. We tried pruning the search tree in various ways, such as remembering only the 10 best paths so far found, but without significant improvement. It appeared that it was necessary for the algorithm to consider each point at least once, and consequently, the bookkeeping involved in the A* algorithm was a major source of inefficiency. Accordingly, we developed a more efficient, pseudo-parallel search algorithm.

The pseudo-parallel algorithm is iterative, with some of the characteristics of relaxation processes. It operates by maintaining an array of path costs, one for each image point, representing the path cost for the best path so far found from the start point to the image point in question. The path cost for each point is computed by examining its eight neighbors and determining which one currently has

the minimal path cost. The path cost for the point under consideration is then the minimal neighboring path cost plus the step cost for stepping to the corresponding neighbor. The path cost for the start point is always zero.

The path cost computation is iterated over the whole array until no path cost value changes, at which point we have an array that contains the true minimal path cost for every point. The iteration is performed by scanning the points in a raster scan and computing an updated cost for each. Information propagates rapidly with the direction of scan, but slowly against it. We therefore scan in four different directions, cycling through them until no change occurs in a scan. It should be clear that the iterative scan is a serial implementation of what is essentially a parallel computation, a computation which , by its simplicity, suggests itself as a candidate for implementation in integrated circuitry.

It is straightforward to actually determine the minimal cost path from any point to the start: by simply always stepping to the neighboring point whose path cost is least. Thus, we can readily find the road between two points in the image, despite local obscurations. Moreover, since the pseudo-parallel search algorithm finds all best paths from the start point, when the end point is uncertain we simply look in a small area for the minimal cost path whose cost is least. The pseudo-parallel search algorithm is faster than the A* algorithm; however, we would like to improve its speed even further. We have observed that in an area in which the road is well-defined, the corresponding section of path is dominant in searching. That is, many minimal cost paths from nearby points run directly to the dominant path and then along it. In such cases, the original road operator evidence is strong enough for a much simpler algorithm to trace. We have experimented with an elementary algorithm that steps along the path, at each point stepping in the direction with least cost, so long as that cost is significantly lower than the next best. When the issue is not clear cut, the algorithm resorts to a very limited lookahead of just a

few steps: if none of the extensions of the path is clearly best, it resorts to calling the pseudo-parallel search algorithm. Preliminary results indicate that it may be possible to speed up the tracing process considerably for well-defined roads.

d. Guidance

An important consideration in designing the road-finding program was the need to be able to accept guidance, either from the user, or from a priori knowledge, such as a map or previously interpreted image. Guidance can dramatically improve efficiency by eliminating processing of irrelevant information, and improve robustness by preventing distraction by anomalous local context.

In the present system, the guidance is input in the form of an approximate trace of the road in question, either provided by the user tracing the road crudely with a cursor on the display, or provided by a prediction based on the image camera calibration and the representation of the road in the map. The guideline is, in both cases, a chain of straight line segments superimposed on the image.

The program uses the guideline one line segment at a time and creates a box enclosing the segment. The box is larger than the minimal enclosing rectangle to tolerate possible errors in the location of the guideline. In addition, a mask array is created containing a 1 for each pixel within the box whose distance from the guideline is within tolerance, and a 0 for the other pixels. The box thus provides a coarse limitation on the area under consideration, whereas the mask provides a finer limitation.

The road operator is then scanned over the box area and applied where the mask array is 1. The parallel search algorithm is then applied to the results, with the initial point of the guideline as the origin of the search, and a high cost associated with stepping outside the masked area. The best path is found by searching a small area near the end point of the guideline for the path with least cost.

Finally, the best path is traced out, and the process is repeated with the next segment of the guideline.

3. Performance

Figure 21 shows a test image of a rural area with a guideline provided by the user superimposed. In Figure 22, the system has boxed the first segment of the guideline and traced the fragment of road within the box. In Figure 23, all segments of the guideline have been boxed and used to trace the road in entirety. The resulting trace of the road is shown in Figure 24.

Figure 25 shows the result of tracing many of the roads visible in the image. Note that the program has traced the center line of the wide road. Note particularly that it has performed extremely well in areas in which the road is faint or partially obscured, such as at the lower left and the upper right of the image.

The program has also been applied to images of urban areas. Figure 26 shows the results of guided tracing in an area containing many intersecting streets. In Figure 27, the tracings have been fitted with straight line segments to cartographic accuracy. The results here, too, are extremely good.

We have so far performed only a limited number of experiments with interactive road tracing. The results, however, are most encouraging. The system is capable of tracing linear features that are hard even for a human to discern through a wide range of terrain types and environments. It can accept guidance from a human user, or from a preexisting map; it needs very little guidance, but the more guidance it is given, the more reliable and efficient is its performance.

There are many ways we can improve performance even further. The road operator might be improved by underpinning it with more secure theoretical foundations, and hence perhaps determining more appropriate scoring functions and combinations of their results. The scoring functions might also be made adaptable to the particular image or class

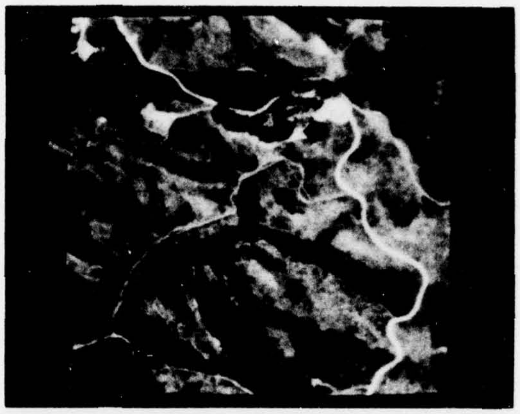


FIGURE 21 A RURAL ROAD WITH A GUIDELINE

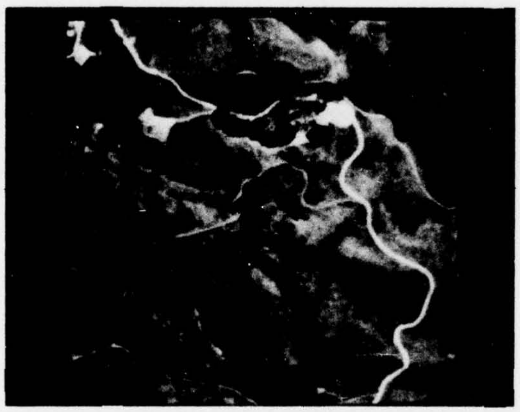


FIGURE 22 TRACING USING A SEGMENT OF GUIDELINE

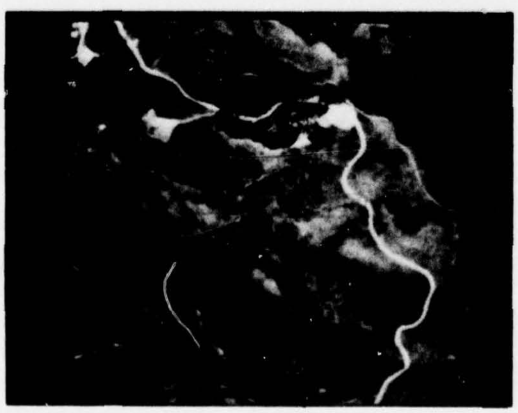


FIGURE 23 GUIDED TRACING COMPLETED

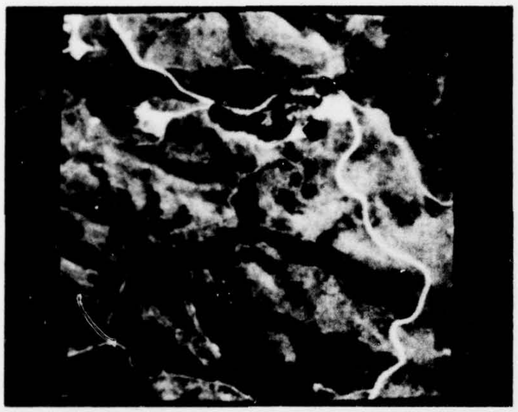


FIGURE 24 RESULT OF TRACING ONE RURAL ROAD

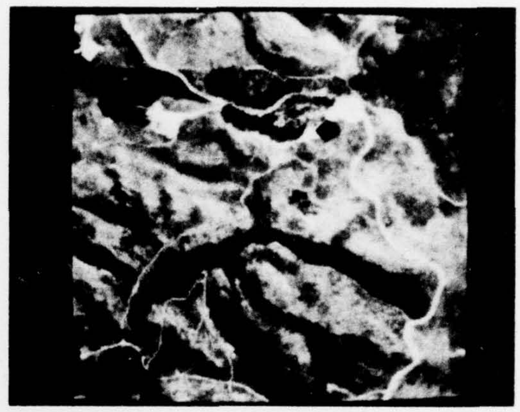


FIGURE 25 GUIDED TRACING OF SEVERAL RURAL ROADS

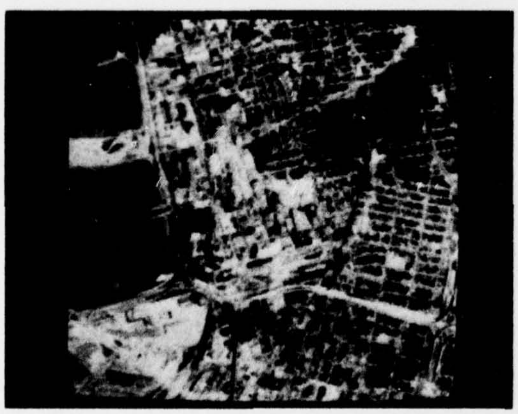


FIGURE 26 GUIDED TRACING OF URBAN STREETS

FIGURE 27 TRACINGS FITTED BY STRAIGHT LINES

of image under consideration. The tracing algorithm can certainly be speeded up by recoding, and in clear-cut cases, when the full power of the pseudo-parallel search is not required, it could be replaced by a simpler, considerably faster, direct tracing algorithm.

An important area for improvement is the tactics and strategy of road finding. The operator we have described, while handling well a variety of road widths and environments, is not a universal operator; it does not cope well with multi-lane highways, or cloverleaf intersections, for example. Such cases are better handled by more specialized processes. A higher level program is required that has many such tactical processes at its disposal, and that can decide which of them is most appropriate for a particular task. Whether to look for dark roads on a light background, or the converse, depends on the conditions under which the image was taken, and is a simple decision that could often be safely left to the system. Performance using an appropriate operator would be better than that using a more general operator. In addition, the present system has no understanding of intersections, bridges, tunnels, and so forth. Embodying knowledge of such features in the system would enable it to handle them more intelligently and hence extend its capabilities to a wider range of situations.

During the next stage of our research program, we shall be attempting to develop expert subsystems that deal with a particular class of task competently. Road finding is clearly an appropriate task for study and in-depth development.

D. Knowledge Representation

1. Overview

The system towards which we are working will need to contain and access a great deal of knowledge about a wide range of subjects. Since our work emphasizes the use of maps as aids to interpretation of images, we wish to store and access quantities of information of the sort contained in conventional maps. In addition, we wish to include other sorts of information relating to features in the map, information that is not conventionally included, such as functions of factories or descriptions of appearance.

In addition to knowledge about the world, it is essential for interpretation that knowledge of the relationship between world and image be available. This knowledge involves the physics of the sensor, the parameters to which it is sensitive, the spatial correspondence between image and world, location of the viewpoint, and the prevailing conditions at the time the image was formed.

Finally, a sophisticated image understanding system must contain knowledge about itself: about the applicability, reliability and efficiency of its components. For example, several different edge detectors may be available within the system, but for the particular task in hand encompassing object, background, and viewing conditions, one detector may perform well and another badly; intelligent selection is required.

We are currently considering the questions of what knowledge must be represented in the system, how it should be represented, and how it should be exploited. Our studies so far are preliminary and are continuing. The current state of our thinking is presented here.

2. Content and Use

A primary function of the knowledge base is to represent and retrieve information concerning the geometry and topology of the world.

Locations of point features, like bridges, must be recorded, together with linear features, like roads, which link them. We need to be able to retrieve answers to questions, such as "Where is factory Y ?" and "What roads pass through UTM grid square Z ? " We also need to be able to record whatever characteristics of a feature we think important, like clearance under bridges.

It is necessary to handle generic information such as "Rivers go under bridges, but not over them" because it can be used to aid the interpretation of the image (for example, an automatic interpretation of the area where a particular road apparently intersects a river). Such information can constrain a search to relevant areas, eliminate impossible interpretations, suggest likely events, and so on.

Internal descriptions of data classes are also valuable for a number of reasons: The system can verify correctness of data given to it and it can solicit and store information when it is required. Moreover, such information facilitates automatic plan formation and provides a form of documentation that is very useful when a number of people are simultaneously working on and improving the system.

3. Representation

We have been experimenting with data structures for the knowledge base. One requirement is that it should be quick to get from one piece of information to a closely related one. Another requirement is that it should be possible to store large quantities of information. The representation scheme we have adopted is a variety of a semantic net.

Each concept or object of interest is represented by a LISP atom, the property list of which contains properties and binary relations linking it to other objects. For example a particular bridge, say B0021, might have as its property list (WIDTH 20 OVERGOER R0932 ...), where R0932 is the atom representing a particular road. Thus a semantic network of objects and relations is

established in which it is easy to step from one object to a related one. In many cases, we may wish to step either from Object A to Object B, or vice versa. We therefore ensure that relations are represented by entries on the property lists of both objects: in our previous example, R0932 might have as its property list (OVERGOER-OF B0021 ...).

Since we also wish to retrieve objects generically, all bridges, say, we include in our semantic network objects that represent sets; elements of a set are linked to it by the ELEMENT relation, and the set is linked to other sets by the SUBSET relation. Other important concepts in the network are partitions of a set--a set of disjoint subsets of that set--and the delineator of a set--a canonical example of an element of that set.

Figure 28 shows a fragment of the network as an example. BRIDGE53 represents a particular bridge that has ROAD970 as its OVERGOER and RIVER102 as its UNDERGOER. (The names are arbitrary and meaningless to the system.) BRIDGE53 is recognizable as a bridge because it is an ELEMENT of the set BRIDGES. The object E.G.BRIDGE is the delineator of BRIDGES, so it is clear that bridges in general have an UNDERGOER that is an element of the set BRIDGE-UNDERGOERS. The latter set has a PARTITION whose subsets are ROADS, RAILS, and RIVERS. We can see from the figure that a river can go under a bridge, but not over it, since RIVERS is not a subset in the partition of BRIDGE-OVERGOERS.

Figure 29 shows an example of part of the subset hierarchy. The entities in the world can be partitioned according to whether they are area-like, point-like, or line-like. Alternatively, they can be partitioned according to whether they are natural or manmade. Area features can be partitioned according to terrain type, or according to the use made of them. If we wished to find all military airfields (assuming we do not already have them explicitly listed) our network tells us that they are particular types of airfield, and that airfields are contained in flat terrain. Thus the system could decide to look for new airfields only in areas known to be flat.

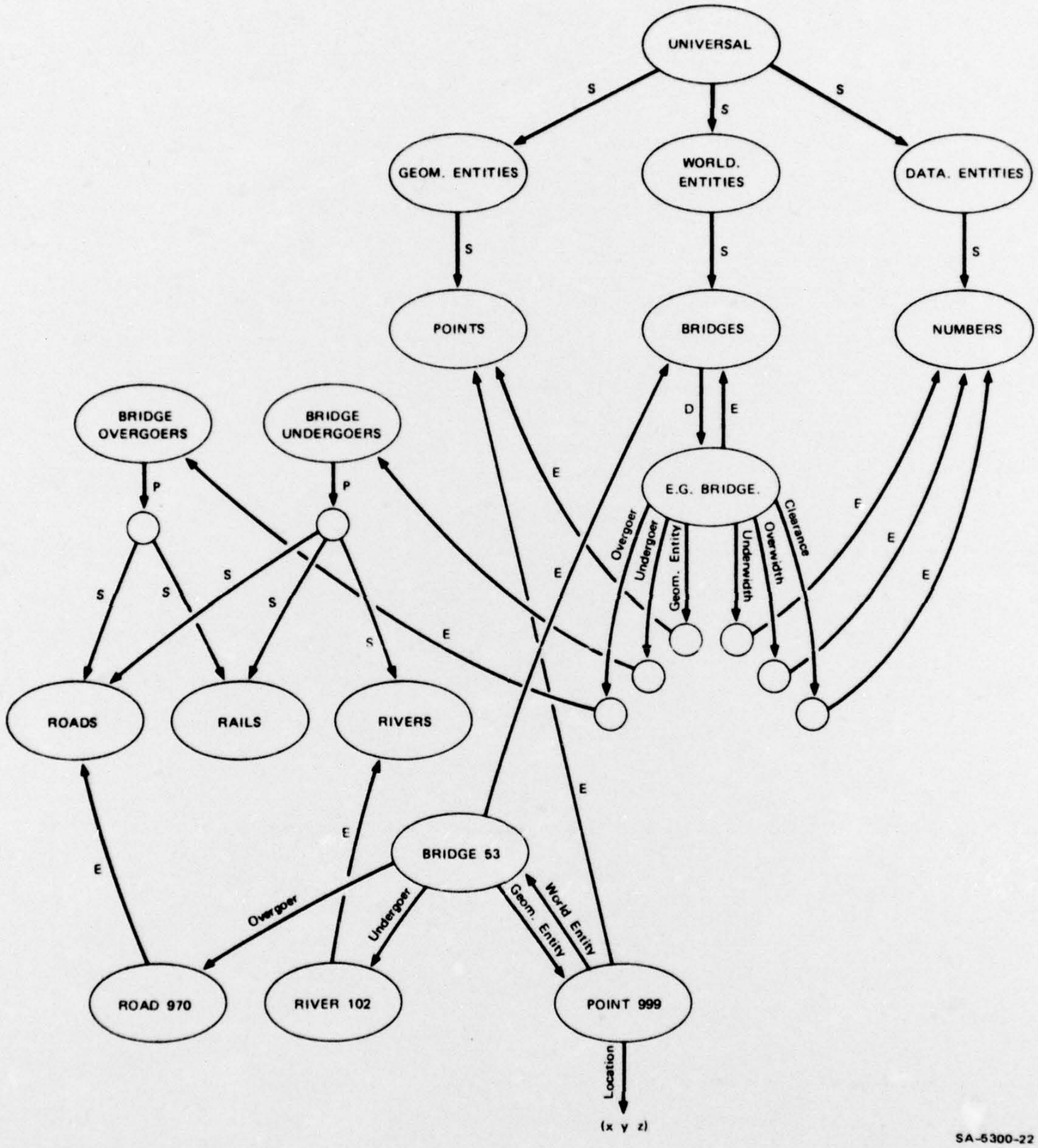


FIGURE 28 A NETWORK DESCRIPTION OF A PARTICULAR BRIDGE

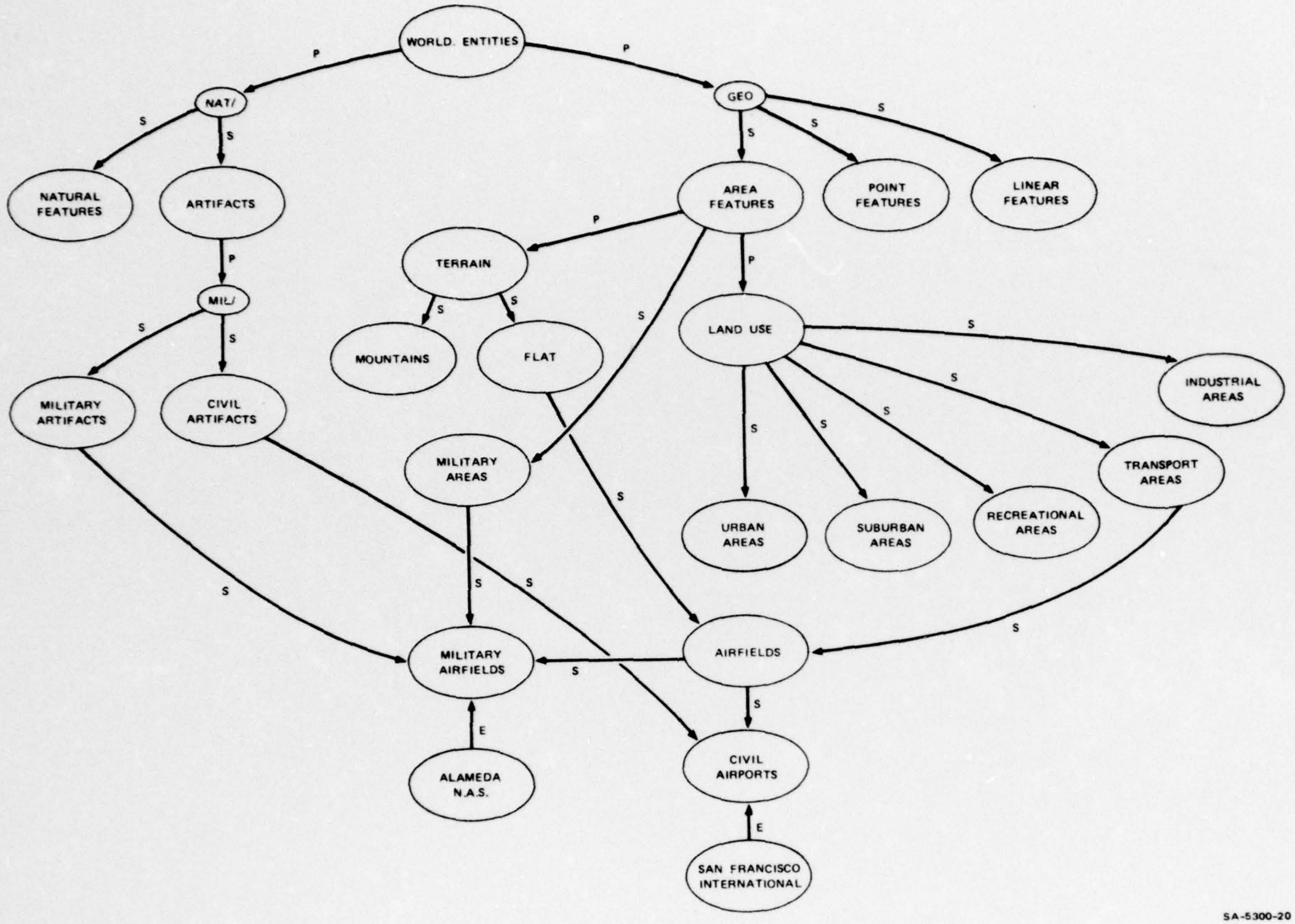


FIGURE 29 PART OF THE NETWORK SUBSET HIERARCHY

World entities have associated geometrical entities--points, lines and regions--that represent them for the purposes of making geometrical or topological inferences. A bridge has an associated point that gives its location; a road segment has an associated line joining two points. This association is indicated in Figure 28 and in Figure 30, the latter of which shows the network fragment describing the geometrical entities. It should be clear that we are now in a position to determine locations of objects and distances between them or to find routes using the geometrical entities.

It is necessary to be able to determine what is to be found in a particular part of the world, so some sort of geometrical index into the data is required. We are experimenting with a hierarchical index as follows: We consider the world to be comprised of rectangular cells whose corners have known coordinates. Points, lines, and regions that are totally enclosed by the cell are associated with the cell. If too much information is associated with one cell, the cell is partitioned into subcells and the information is associated with the subcells as far as possible: a line that passes from one subcell to another must be associated with the parent cell rather than with the subcells. The cells thus form a hierarchical structure with one cell at the top encompassing the entire area covered by the data base (see Figure 31).

We can readily determine which objects are contained in a specified area by stepping down through the hierarchy. We consider only subcells that overlap the area, and then only their subcells that overlap it, and so on. The process is strongly focused onto only the bottom-most cells that overlap the area in question. Similar algorithms have been used for hidden line removal in line drawing generation; they have proved to be fast and efficient.

Finally, in the network we also have descriptions of the different conceptual data types in the system, such as pictures, maps, transforms, and displays, together with their interrelationships. This enables us to have documentation of the data types in use permanently

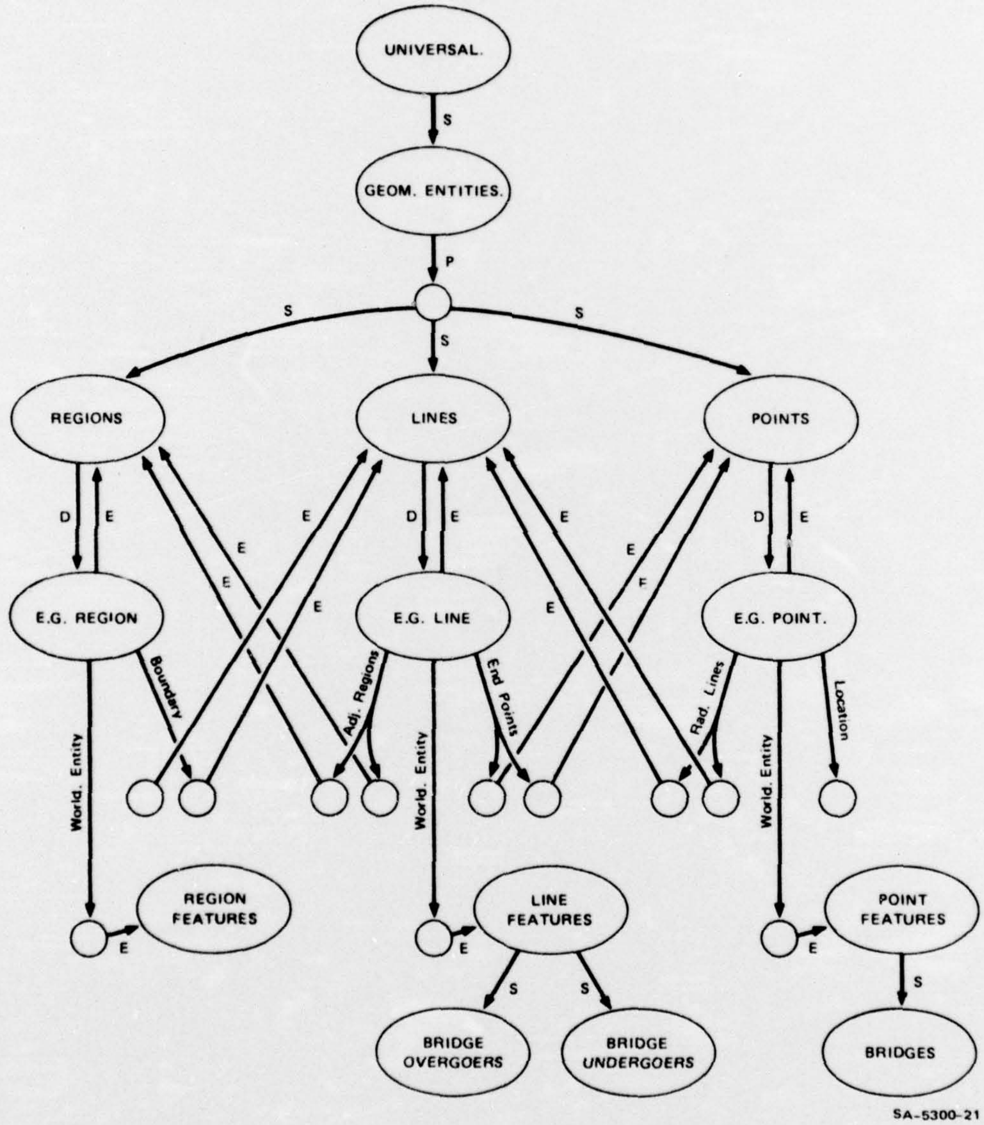


FIGURE 30 NETWORK REPRESENTATION OF GEOMETRY AND TOPOLOGY

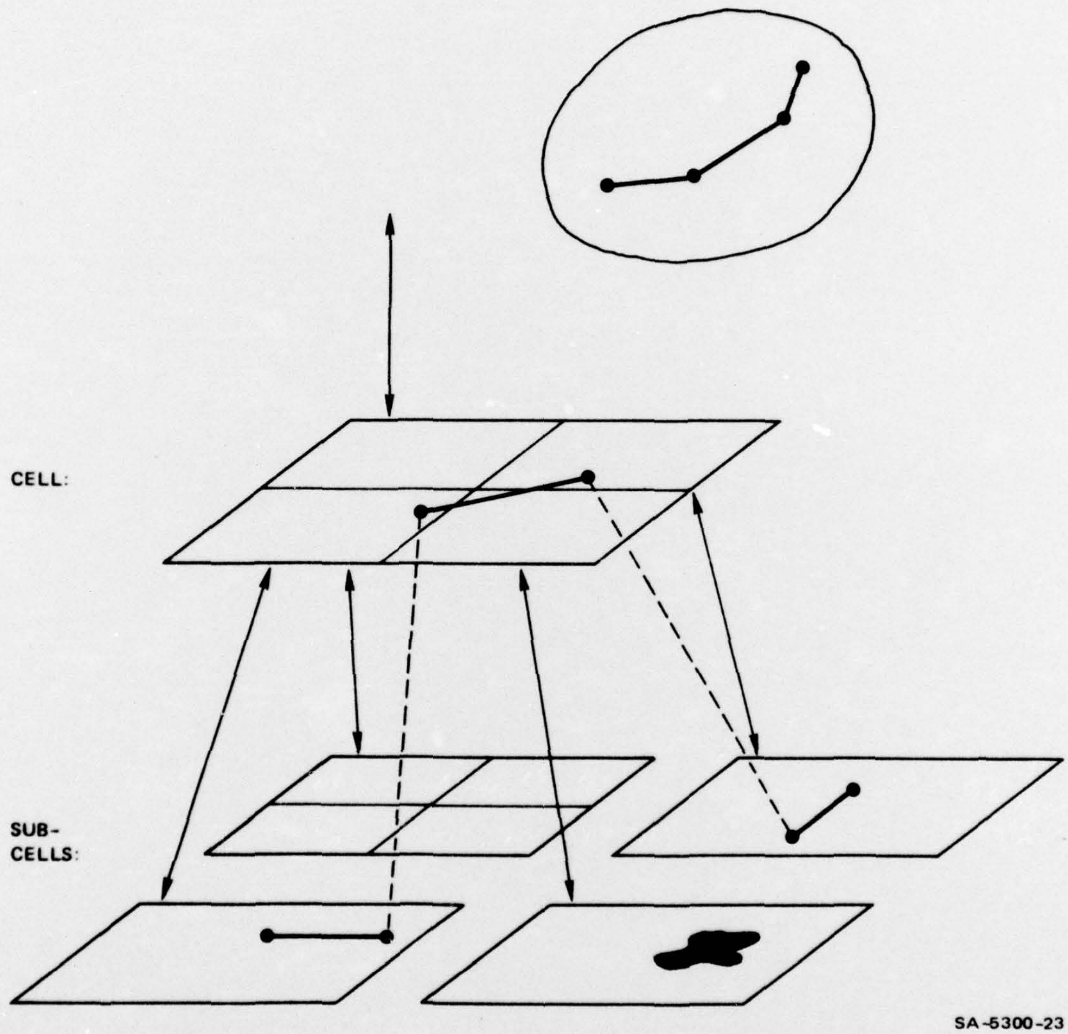


FIGURE 31 HIERARCHICAL GEOMETRIC INDEX

available to the user and to the system. We plan to include descriptions of the system's functions to facilitate intelligent strategy planning for image understanding.

4. Quantity data storage

We expect to be dealing with very large quantities of information in the knowledge base, far too much to be maintained completely in a data structure in core. We have therefore designed our representation so that fragments of the network may be read in from disc file, examined or modified in core, and rewritten onto disc if necessary.

We maintain a buffer in core of perhaps up to 1000 objects (atoms), each of which may mention in its property list other objects that may or may not also be in core. Each atom is tagged when its properties are read from disc. Thus, when we wish to step from one object to a related one, the system can check whether the related object is in core. If it is not, it can be read in from disc, possibly after writing out another object to make room in the buffer. The buffer is ordered by time of use, so objects used least recently can be discarded first. Objects are rewritten onto disc only if they have been modified during their period in core.

This data base maintenance scheme has the advantage of permitting storage of arbitrarily structured information, rather than fixed-format records. It remains to be seen how well it performs with very large quantities of data, but since we anticipate that the system will perform a lot of computation with each chunk of data it reads in, we expect it to provide reasonably good support for our work.

III WORK PLANNED FOR THE NEXT SIX MONTHS

During the past six months, we have adhered to the research plan laid out in the project proposal. We have encountered no insurmountable problems, and our work is very much on course. We intend to continue following the research plan, so this section will be brief.

The plan calls for us to continue exploratory investigations into a wide range of tasks and techniques, but to begin transferring effort to the development of expert sub-systems. We have progressed more rapidly than expected in the area of system development, and have capitalized on the time gained by pursuing some of our explorations in considerable depth. We are therefore in a strong position for the next stage of research.

The problem of representation is central, and one to which we feel it is important to direct part of our effort. We therefore intend to pursue our current approach in this area and to explore ways in which the knowledge base can be employed to direct the lower level processes more effectively.

So far, we have two strong candidates for expert sub-systems: map/picture correspondence, and linear feature tracing. The first task is fundamental, in that map/picture correspondence is important for intelligent image analysis and understanding. It has important application in registering images from multiple sensors, in navigation, map updating, and photo interpretation. Our present system relies on interaction to determine correspondence, but we can certainly automate the process in many cases, leading to the capability for totally automatic monitoring.

The second task, linear feature tracing, is important for such applications as map updating and photo interpretation. A tracing

subsystem could be a valuable component in establishing map/picture correspondence. We have been successful in tracing roads in particular situations, and it would be appropriate to capitalize on what we have already achieved, extending the scope of the existing system considerably.

REFERENCES

1. "Artificial Intelligence -- Research and Applications," Annual Report, SRI Projects 3805 and 4763, Ed. B. Raphael, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, CA. 94025, June 1976.

2. M. Hueckel, "An Operator which Locates Edges in Digital Pictures," Journal of the ACM, Vol. 18, No. 1, January 1971.

3. R.O. Duda, and P.E. Hart, "Pattern Classification and Scene Analysis," Wiley, 1973.